



15
AFB

PATENT
Customer No. 22,852
Attorney Docket No. 07643.0042-00

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re Application of:)	
)	
Russell T. DAVIS et al.)	Group Art Unit: 2176
)	
Application No.: 10/052,250)	
)	Examiner: C. Nguyen
Filed: January 23, 2002)	
)	
For: RDX ENHANCEMENT OF)	
SYSTEM AND METHOD FOR)	Confirmation No.: 1920
IMPLEMENTING REUSABLE)	
DATA MARKUP LANGUAGE)	
(RDL))	

Attention: Mail Stop Appeal Brief - Patents
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

Sir:

APPEAL BRIEF UNDER BOARD RULE § 41.37

In support of the Notice of Appeal filed January 31, 2008, and further to 37 C.F.R. 41.37(a)(1), Appellants present this brief and enclose herewith a check for the fee of \$510.00 required under 37 C.F.R. 41.20(b)(2). Appellants also include a Petition for Extension of Time of one month to extend the time period for responding to the Notice of Panel Decision from Pre-Appeal Brief Review to September 2, 2008 (August 30, 2008 being a Saturday, August 31, 2008 being a Sunday, and September 1, 2008 being a Federal Holiday).

08/29/2008 SZEWDIE1 00000074 10052250

01 FC:1402

510.00 0P

Customer No. 22,852
Application No.: 10/052,250
Attorney Docket No. 07643.0042-00

This Appeal responds to the final rejection of claims 1-6, 8-21, 23-34, and 36-64
in the Final Office Action mailed November 1, 2007.

If any additional fees are required or if the enclosed payment is insufficient,
Appellants request that the required fees be charged to Deposit Account No. 06-0916.

TABLE OF CONTENTS

I. Real Party in Interest	4
II. Related Appeals and Interferences.....	5
III. Status of Claims.....	6
IV. Status of Amendments	6
V. Summary of Claimed Subject Matter.....	8
VI. Grounds of Rejection.....	14
VII. Argument.....	15
VIII. Claims Appendix to Appeal Brief Under Rule 41.37(c)(1)(viii)	23
IX. Evidence Appendix to Appeal Brief Under Rule 41.37(c)(1)(ix).....	36
X. Related Proceedings Appendix to Appeal Brief Under Rule 41.37(c)(1)(x)	37

I. Real Party in Interest

e-Numerate Solutions, Inc. is the real party in interest.

II. Related Appeals and Interferences

There are currently no other appeals or interferences, of which Appellants, Appellants' legal representative, or assignee are aware, that will directly affect or be directly affected by or have a bearing on the Board's decision in the pending appeal.

III. Status of Claims

A. Claims 62-64 stand rejected under 35 U.S.C. § 103(a) as being unpatentable over U.S. Patent No. 6,721,736 to Krug et al. (*Krug*) in view of the "XBRL Specification" by Hamscher et al. (*Hamscher*).

B. Claims 1-6, 11-21, 24-34, 37-46, 49-57, and 59-61 stand rejected under 35 U.S.C. § 103(a) as being unpatentable over U.S. Patent No. 6,370,549 to Saxton (*Saxton*), in view of U.S. Patent Application Publication No. 2002/0052954 to Polizzi et al. (*Polizzi*), and further in view of *Hamscher*.

C. Claims 8-10, 23, 36, 47, 48, and 58 stand rejected under 35 U.S.C. § 103(a) as being unpatentable over *Saxton*, *Polizzi*, *Hamscher*, and further in view of U.S. Patent No. 6,134,563 to Clancey et al. (*Clancey*).

D. Claims 7, 22, and 35 have been canceled.

Appellants appeal the rejection of claims 1-6, 8-21, 23-34, and 36-64. The attached Appendix contains a clean copy of these claims.

IV. Status of Amendments

No amendments have been filed subsequent to the final rejection of claims 1-6, 8-21, 23-34, and 36-64 in the Final Office Action mailed November 1, 2007.

V. Summary of Claimed Subject Matter

Independent claims 1, 17, 29, 30, 42, 54, and 62 recite a method, apparatus, and system for processing data and developing a report.

Independent claim 1 is directed to a data processing system for developing a report. *See*, for example, specification at page 17, paragraphs 056-057. The system includes a parser that receives one or more text documents and creates software elements having a format with a hierarchal relationship between the software elements based on the one or more text documents. *See*, for example, specification at page 18, paragraph 059 and Fig. 2, ref. 204. The system includes a manager that provides for the creation of a second hierarchical relationship between the software elements and the restructuring of the first hierarchical relationship and the second hierarchical relationship into software structures corresponding to a new text document. *See*, for example, specification at page 18, paragraph 059 and Fig. 2, ref. 206. The system also includes an editor that develops a report by referencing the software elements created from the one or more text documents to form a structure of the report and retrieves data from one or more sources to represent one or more values within the report. *See*, for example, specification at pages 18-19, paragraph 060 and Fig. 2, ref. 208. The system further includes a mapper that generates a relationship between the data from the one or more sources and the one or more values to be placed within the report. *See*, for example, specification at pages 18-19, paragraphs 0062-0063 and Fig. 2, ref. 210.

Independent claim 17 is directed to a method in a data processing system for developing a report. *See*, for example, specification at page 17, paragraphs 056-057. The method includes receiving one or more text documents. *See*, for example,

specification at page 18, paragraph 059. The method includes creating software elements having a format with a hierarchal relationship between the software elements based on the one or more text documents. See, for example, specification at page 18, paragraph 059. The method includes creating a second hierarchical relationship between the software elements. See, for example, specification at page 18, paragraph 059. The method also includes restructuring the first hierarchical relationship and second hierarchical relationship into software structures corresponding to a new text document. See, for example, specification at page 18, paragraph 059. The method also includes developing a report by referencing the software elements created from the one or more text documents to form a structure of the report and retrieving data from one or more sources to represent one or more values within the report. See, for example, specification at pages 18-19, paragraph 060. The method further includes generating a relationship between the data from one or more sources and the one or more values to be placed within the report. See, for example, specification at pages 18-19, paragraphs 0062-0063.

Independent claim 29 is directed to a data processing system for developing a report. See, for example, specification at page 17, paragraphs 056-057. The system includes a means for receiving one or more text documents. See, for example, specification at page 18, paragraph 059 and Fig. 2, ref. 204. Exemplary structure corresponding to the means for receiving one or more text documents is RDX parser 204. The system includes a means for creating software elements having a format with a hierarchal relationship between the software elements based on the one or more text documents. See, for example, specification at page 18, paragraph 059 and Fig. 2, ref.

204. Exemplary structure corresponding to the means for creating software elements is RDX parser 204. The system includes a means for creating a second hierarchical relationship between the software elements. See, for example, specification at page 18, paragraph 059 and Fig. 2, ref. 206. Exemplary structure corresponding to the means for creating a second hierarchical relationship is RDX manager 206. The system also includes a means for restructuring the first hierarchical relationship and the second hierarchical relationship into software structures corresponding to a new text document. See, for example, specification at page 18, paragraph 059 and Fig. 2, ref. 206.

Exemplary structure corresponding to the means for restructuring the first hierarchical relationship and the second hierarchical relationship is RDX manager 206. The system also includes a means for developing a report by referencing the software elements created from the one or more text documents to form a structure of the report and retrieving data from one or more sources to represent one or more values within the report. See, for example, specification at pages 18-19, paragraph 060 and Fig. 2, ref.

208. Exemplary structure corresponding to the means for developing a report is RDX document editor 208. The system further includes a means for generating a relationship between the data from the one or more sources and the one or more values to be placed within the report. See, for example, specification at pages 18-19, paragraphs 0062-0063 and Fig. 2, ref. 210. Exemplary structure corresponding to the means for generating a relationship is RDX mapper 210.

Independent claim 30 is directed to a computer-readable medium including instructions for controlling a processor to perform a method for developing a report. See, for example, specification at page 17, paragraphs 056-057. The method includes

receiving one or more text documents. *See*, for example, specification at page 18, paragraph 059. The method includes creating software elements having a format with a hierarchal relationship between the software elements based on the one or more text documents. *See*, for example, specification at page 18, paragraph 059. The method includes creating a second hierarchical relationship between the software elements. *See*, for example, specification at page 18, paragraph 059. The method also includes restructuring the first hierarchical relationship and the second hierarchical relationship into software structures corresponding to a new text document. *See*, for example, specification at page 18, paragraph 059. The method also includes developing a report by referencing the software elements created from the one or more text documents to form a structure of the report and retrieving data from one or more sources to represent one or more values within the report. *See*, for example, specification at pages 18-19, paragraph 060. The method further includes generating a relationship between the data from the one or more sources and the one or more values to be placed within the report. *See*, for example, specification at pages 18-19, paragraphs 0062-0063.

Independent claim 42 is directed to a data processing system for developing a report. *See*, for example, specification at page 17, paragraphs 056-057. The system includes a parser that receives one or more text documents and creates software elements having a format with a hierarchal relationship between the software elements based on the one or more text documents. *See*, for example, specification at page 18, paragraph 059 and Fig. 2, ref. 204. The system includes a manager that manipulates the software elements, provides for the creation of a second hierarchical relationship between the software elements, and provides for the restructuring of the first

hierarchical relationship and the second hierarchical relationship into software structures corresponding to a new text document. *See*, for example, specification at page 18, paragraph 059 and Fig. 2, ref. 206. The system also includes an editor that develops a report by referencing the software elements created from the one or more text documents to form a structure of the report. *See*, for example, specification at pages 18-19, paragraph 060 and Fig. 2, ref. 208. The system further includes a mapper that retrieves data from one or more sources to represent one or more values within the report and generates a relationship between the retrieved data and the one or more values within the report. *See*, for example, specification at pages 18-19, paragraphs 0062-0063 and Fig. 2, ref. 210.

Independent claim 54 is directed to a method for data processing. *See*, for example, specification at page 17, paragraphs 056-057. The method includes receiving one or more text documents. *See*, for example, specification at page 18, paragraph 059. The method includes creating software elements having a format with a hierarchical relationship between the software elements based on the one or more text documents. The method includes manipulating the software elements. *See*, for example, specification at page 18, paragraph 059. The method also includes creating a second hierarchical relationship between the software elements. *See*, for example, specification at page 18, paragraph 059. The method also includes restructuring the first hierarchical relationship and the second hierarchical relationship into software structures corresponding to a new text document. *See*, for example, specification at page 18, paragraph 059. The method also includes developing a report by referencing the software elements created from the one or more text documents to form a structure

of the report. *See*, for example, specification at pages 18-19, paragraph 060. The method also includes generating a relationship between data from one or more sources and one or more values to be placed within the report. The method further includes retrieving data from the one or more sources to represent the one or more values within the report. *See*, for example, specification at pages 18-19, paragraphs 0062-0063.

Independent claim 62 is directed to a data processing system. *See*, for example, specification at page 17, paragraphs 056-057. The system includes a parser that receives one or more text documents. *See*, for example, specification at page 18, paragraph 059 and Fig. 2, ref. 204. The parser interprets tags included in the one or more text documents to create software elements. *See*, for example, specification at page 18, paragraph 059 and Fig. 2, ref. 204. The parser also determines the hierarchy of the software elements within a structure representative of the one or more text documents. *See*, for example, specification at page 18, paragraph 059 and Fig. 2, ref. 204. The system also includes a manager that provides for the creation of a second hierarchy of the software elements. *See*, for example, specification at page 18, paragraph 059 and Fig. 2, ref. 206. The manager also provides for the restructuring of the first hierarchy and the second hierarchy into software structures corresponding to a new text document. *See*, for example, specification at page 18, paragraph 059 and Fig. 2, ref. 206.

VI. Grounds of Rejection

A. Whether claims 62-64 are unpatentable under 35 U.S.C. § 103(a) over U.S. Patent No. 6,721,736 to Krug et al. (*Krug*) in view of the "XBRL Specification" by Hamscher et al. ("*Hamscher*").

B. Whether claims 1-6, 11-21, 24-34, 37-46, 49-57, and 59-61 are unpatentable under 35 U.S.C. § 103(a) over U.S. Patent No. 6,370,549 to Saxton ("*Saxton*"), in view of U.S. Patent Application Publication No. 2002/0052954 to Polizzi et al. ("*Polizzi*"), and further in view of *Hamscher*.

C. Whether claims 8-10, 23, 36, 47, 48, and 58 are unpatentable under 35 U.S.C. § 103(a) over *Saxton*, *Polizzi*, *Hamscher*, and further in view of U.S. Patent No. 6,134,563 to Clancey et al. ("*Clancey*").

VII. Argument

A. The rejection of claims 62-64 under U.S.C. § 103 as being unpatentable over *Krug* and *Hamscher* is improper

The Examiner's rejection of claims 62-64 under 35 U.S.C. § 103(a) as being unpatentable over *Krug* and *Hamscher* should be reversed. Appellants submit that a *prima facie* case of obviousness has not been established.

The key to supporting any rejection under 35 U.S.C. § 103 is the clear articulation of the reason(s) why the claimed invention would have been obvious. See M.P.E.P. § 2142, 8th Ed., Rev. 6 (Sept. 2007). Such an analysis should be made explicit and cannot be premised upon mere conclusory statements. See *id.* "A conclusion of obviousness requires that the reference(s) relied upon be enabling in that it put the public in possession of the claimed invention." M.P.E.P. § 2145. Furthermore, "[t]he mere fact that references can be combined or modified does not render the resultant combination obvious unless the results would have been predictable to one of ordinary skill in the art" at the time the invention was made. M.P.E.P. § 2143.01(III), internal citation omitted. Moreover, "[i]n determining the differences between the prior art and the claims, the question under 35 U.S.C. § 103 is not whether the differences themselves would have been obvious, but whether the claimed invention as a whole would have been obvious." M.P.E.P. § 2141.02(I), internal citations omitted (emphasis in original).

In this application, a *prima facie* case of obviousness has not been established because the Examiner has not clearly articulated a reason why one of ordinary skill in the art would find the claimed combination obvious in view of the cited references.

Claim 62 recites a data processing system, comprising:

a parser that:

...

interprets tags included in the one or more text documents to create software elements, and determines the hierarchy of the software elements within a structure representative of the one or more text documents; and

a manager that:

provides for the creation of a second hierarchy of the software elements, and provides for the restructuring of the first hierarchy and the second hierarchy into software structures corresponding to a new text document.

(emphasis added). In *Krug*, a syntax tree parser 20 “analyses the HTML syntax structure of the search result document by recognizing the HTML tags within the document and constructing a hierarchical HTML syntax tree that represents the hierarchical relationship of the syntax elements (tags)” (col. 8, lines 23-27). The Examiner states that the syntax elements in *Krug* constitute the claimed “software elements” (Final Office Action at page 3). This is not correct. Even if the syntax elements of *Krug* could be reasonably construed as some type of “software elements,” which Appellants do not concede, they cannot constitute the claimed “software elements,” which are created by interpreting “tags” in the “text documents.”

Krug specifically teaches that the syntax elements are the “tags” within the document (col. 8, line 27). By alleging that the syntax elements in *Krug* could somehow constitute the claimed “software elements,” the Examiner is asserting that the tags in *Krug* correspond to both the claimed “tags” and the claimed “software elements.” Therefore, according to the Examiner’s statements, *Krug* interprets tags included in the document to create tags. This is not correct.

Krug analyzes the HTML syntax structure by recognizing tags and constructs a syntax tree that represents the hierarchical relationship of the tags. Neither the tags, syntax elements, nor any other teaching in *Krug* constitute the claimed “software elements” at least because *Krug* does not interpret “tags included in the one or more text documents to create software elements,” as recited in claim 62. Accordingly, *Krug* also cannot teach or suggest determining “the hierarchy of the software elements within a structure representative of the one or more text documents,” as further recited in claim 62.

Even assuming that the “hierarchical structure” of *Krug* could somehow correspond to the claimed “hierarchy of the software elements,” which Appellants do not concede, the Examiner correctly states that *Krug* “does not explicitly disclose a manager that provides for the creation of a second hierarchy . . .” (Final Office Action at page 3). The Examiner relies on *Hamscher* to allegedly disclose these elements. However, this is not correct.

Hamscher is directed to a XBRL specification for defining XBRL elements and attributes that can be used in the creation, exchange, and comparison tasks of financial reporting (Abstract). The first paragraph on page 17 of *Hamscher* discloses “[o]rder independence also simplifies the combination of financial information from different periods or entities, or even for the same entity under different reporting regimes, since in most cases an XBRL instance document can be created by concatenating other XBRL instance documents.”

According to this passage, an XBRL instance document can be created by concatenating other XBRL instance documents. The Examiner appears to assert that

an XBRL document created by concatenating other XBRL instance documents constitutes the claimed “second hierarchy of software elements.” Even assuming that this newly created document could correspond to a “hierarchy of software elements,” which Applicants do not concede, only one “hierarchy of software elements” would be created (i.e. the created XBRL document).

Both *Krug* and *Hamscher* disclose, at most, information in a single hierarchy (allegedly the hierarchical relationship in *Krug* and the created XBRL document in *Hamscher*). In contrast, claim 62 requires both the determination of a “hierarchy of the software elements” created by interpreting tags included in the one or more text documents” and “the creation of a second hierarchy of the software elements” (emphasis added). The cited references do not provide for both the determination of a “hierarchy of the software elements” and the creation of a “second hierarchy” of the same “software elements,” as required by claim 62.

Therefore, *Hamscher* does not teach or suggest the claimed “creation of a second hierarchy of the software elements.” Accordingly, *Hamscher* does not teach or suggest a manager that “provides for the creation of a second hierarchy between the software elements, and provides for the restructuring of the first hierarchy and the second hierarchy into software structures corresponding to a new text document,” as recited in claim 62.

As set forth above, and contrary to the assertions of the Examiner, the combination of *Krug* and *Hamscher* does not teach or suggest all elements of claim 62. In view of this mischaracterization of the references, the Office Action has neither properly determined the scope and content of the prior art nor properly ascertained the

differences between the prior art and the claimed invention. Therefore, no reason has been clearly articulated as to why the claim would have been obvious to one of ordinary skill in view of the prior art and a *prima facie* case of obviousness has not been established.

Claim 62 is allowable for at least these reasons, and claims 63 and 64 are also allowable at least due to their depending from claim 62.

Therefore, Appellants respectfully request that the Board reverse the rejection of these claims under 35 U.S.C. § 103(a).

B. The rejection of claims 1-6, 11-21, 24-34, 37-46, 49-57, and 59-61 under 35 U.S.C. § 103(a) as being unpatentable *Saxton Polizzi*, and *Hamscher* is improper

The Examiner's rejection of claims 1-6, 11-21, 24-34, 37-46, 49-57, and 59-61 under 35 U.S.C. § 103(a) as being unpatentable over *Saxton Polizzi*, and *Hamscher* should be reversed. Appellants submit that a *prima facie* case of obviousness has not been established because the Examiner has not clearly articulated a reason why one of ordinary skill would find the claimed combination obvious in view of the cited references.

Regarding independent claim 1, the Examiner correctly states that *Saxton* and *Polizzi* "do not explicitly disclose a manager that provides for the creation of a second hierarchy . . ." (Final Office Action at page 6). The Examiner again relies on *Hamscher* to allegedly disclose this element. This is not correct.

As established above, *Hamscher* does not teach the claimed "creation of a second hierarchy of the software elements." Thus, *Hamscher* cannot teach or suggest a "manager that provides for the creation of a second hierarchy between the software

elements and the restructuring of the first hierarchy and the second hierarchy into software structures corresponding to a new text document,” as recited in claim 1.

Accordingly, and contrary to the assertions of the Examiner, the combination of *Saxton*, *Polizzi*, and *Hamscher* does not teach or suggest all elements of claim 1. In view of this mischaracterization of the references, the Office Action has neither properly determined the scope and content of the prior art nor properly ascertained the differences between the prior art and the claimed invention. Therefore, no reason has been clearly articulated as to why the claim would have been obvious to one of ordinary skill in view of the prior art and a *prima facie* case of obviousness has not been established with respect to claim 1.

Claim 1 is allowable for at least these reasons, and claims 2-6, and 11-16 are also allowable for at least the same reasons as claim 1. Independent claims 17, 29, 30, 42, and 54, though of different scope from claim 1, recite limitations similar to those set forth above with respect to claim 1. Claims 17, 29, 30, 42, and 54 are therefore allowable for at least the reasons presented above. Claims 18-21, 24-29, 31-34, 37-41, 43-46, 49-53, 55-57, and 59-61 are also allowable at least due to their dependence from claims 17, 29, 30, 42, and 54 respectively.

Therefore, Appellants respectfully request that the Board reverse the rejection of these claims under 35 U.S.C. § 103(a).

C. The rejection of claims 8-10, 23, 36, 47, 48, and 58 under 35 U.S.C. § 103(a) as being unpatentable over *Saxton*, *Polizzi*, *Hamscher*, and *Clancey* is improper

Regarding the rejection of claims 8-10, 23, 36, 47, 48, and 58, which depend from claims 1, 17, 30, 42, and 54, the Examiner relies on *Clancey* for allegedly

disclosing “a user [that] can create and edit a report, which is created based upon a predefined template” (Final Office Action at page 12). Even assuming this allegation is correct, which Appellants do not concede, *Clancey* fails to cure the deficiencies of *Saxton*, *Polizzi*, and *Hamscher* discussed above. *Clancey* discloses a “method of creating and editing a document containing one or more terms” (col. 1, lines 42-43). *Clancey* does not teach or suggest a “manager that provides for the creation of a second hierarchy between the software elements and the restructuring of the first hierarchy and the second hierarchy into software structures corresponding to a new text document,” as recited in claim 1, similarly recited in independent claims 17, 30, 42, and 54, and required by dependent claims 8-10, 23, 36, 47, 48, and 58.

As explained above, and contrary to the assertions of the Examiner, the elements of claims 1, 17, 30, 42, and 54 are neither taught nor suggested by the prior art references, whether taken individually or in combination. Furthermore, as outlined above, the Examiner has neither properly determined the scope and content of the prior art nor properly ascertained the differences between the prior art and the claimed invention. Therefore, no reason has been clearly articulated as to why the claim would have been obvious to one of ordinary skill in view of the prior art and a *prima facie* case of obviousness has not been established. Claims 8-10, 23, 36, 47, 48, and 58 are also allowable over *Saxton*, *Polizzi*, *Hamscher*, and *Clancey*, for at least the same reasons as claims 1, 17, 30, 42, and 54.

Therefore, Appellants respectfully request that the Board reverse the rejection of these claims under 35 U.S.C. § 103(a).

D. Conclusion

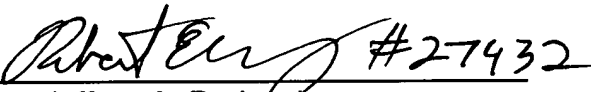
For the reasons given above, pending claims 1-6, 8-21, 23-34, and 36-64 are allowable and reversal of the Examiner's rejections is respectfully requested.

To the extent any extension of time under 37 C.F.R. § 1.136 is required to obtain entry of this Appeal Brief, such extension is hereby respectfully requested. If there are any fees due under 37 C.F.R. §§ 1.16 or 1.17 which are not enclosed herewith, including any fees required for an extension of time under 37 C.F.R. § 1.136, please charge such fees to our Deposit Account No. 06-0916.

Respectfully submitted,

FINNEGAN, HENDERSON, FARABOW,
GARRETT & DUNNER, L.L.P.

Dated: August 28, 2008

By:  #27432
for Jeffrey A. Berkowitz
Reg. No. 36,743

VIII. Claims Appendix to Appeal Brief Under Rule 41.37(c)(1)(viii)

1. A data processing system for developing a report, comprising:
 - a parser that receives one or more text documents and creates software elements having a format with a hierarchal relationship between the software elements based on the one or more text documents;
 - a manager that provides for the creation of a second hierarchical relationship between the software elements and the restructuring of the first hierarchical relationship and the second hierarchical relationship into software structures corresponding to a new text document;
 - an editor that develops a report by referencing the software elements created from the one or more text documents to form a structure of the report and retrieves data from one or more sources to represent one or more values within the report; and
 - a mapper that generates a relationship between the data from the one or more sources and the one or more values to be placed within the report.
2. The data processing system of claim 1, wherein the format with the hierarchal relationship between the software elements is a Numerator Document Object Model (NDOM).
3. The data processing system of claim 1, wherein the one or more text documents are XBRL documents.

4. The data processing system of claim 1, wherein the parser creates the software elements having the format with the hierarchal relationship by interpreting tags included in the one or more text documents.

5. The data processing system of claim 1, wherein a manager manipulates the software elements.

6. The data processing system of claim 5, wherein the manager manipulates the software elements by browsing, editing, loading, and storing the software elements.

8. The data processing system of claim 1, wherein one or more templates are used to develop the report.

9. The data processing system of claim 8, wherein the one or more templates contain data that is directly inserted into the report and instructions enabling data from the one or more sources to be inserted into the report.

10. The data processing system of claim 9, wherein the one or more templates provide instructions to the mapper to retrieve the data that is directly inserted into the report and data from local or remote sources.

11. The data processing system of claim 1, wherein the mapper links the report and the one or more sources that will present one or more values within the report.

12. The data processing system of claim 11, wherein the report and the one or more sources are linked through a “drag and drop” process.

13. The data processing system of claim 1, wherein the editor provides for the software elements to be modified to create a new combination of software elements representative of a new text document.

14. The data processing system of claim 1, wherein the editor provides for modification of one or more parameters associated with the software elements.

15. The data processing system of claim 1, wherein the software elements are transformed to new software elements and are imported into an RDL system.

16. The data processing system of claim 15, wherein the software elements are transformed to the new software elements by retrieving a tag associated with each of the software elements in a dictionary and invoking a translation routine associated with the tag.

17. A method in a data processing system for developing a report, comprising:
receiving one or more text documents;

creating software elements having a format with a hierarchal relationship between the software elements based on the one or more text documents;
creating a second hierarchical relationship between the software elements;
restructuring the first hierarchical relationship and second hierarchical relationship into software structures corresponding to a new text document;
developing a report by referencing the software elements created from the one or more text documents to form a structure of the report and retrieving data from one or more sources to represent one or more values within the report; and
generating a relationship between the data from one or more sources and the one or more values to be placed within the report.

18. The method of claim 17, wherein creating the software elements from the one or more text documents includes representing the software elements in a Numerator Document Object Model (NDOM).

19. The method of claim 17, wherein creating the software elements includes creating software elements from one or more XBRL documents.

20. The method of claim 17, wherein creating the software elements having the format with the hierarchal relationship includes interpreting tags included in the one or more text documents.

21. The method of claim 17, further comprising manipulating the software elements by browsing, editing, loading, and storing the software elements.

23. The method of claim 17, further comprising developing the report from one or more templates, which contain data that is directly inserted into the report and instructions enabling data from the one or more sources to be inserted into the report.

24. The method of claim 22, wherein the relationship is generated through a "drag and drop" process.

25. The method of claim 17, further comprising modifying the software elements to create a new combination of software elements representative of a new text document.

26. The method of claim 17, further comprising modifying the software elements by editing one or more parameters associated with the software elements.

27. The method of claim 17, further comprising transforming the software elements to new software elements for importing into an RDL system.

28. The method of claim 27, wherein transforming the new software elements includes retrieving a tag associated with each of the software elements in a dictionary and invoking a translation routine associated with the tag.

29. A data processing system for developing a report, comprising:

- means for receiving one or more text documents;
- means for creating software elements having a format with a hierarchal relationship between the software elements based on the one or more text documents;
- means for creating a second hierarchical relationship between the software elements;
- means for restructuring the first hierarchical relationship and the second hierarchical relationship into software structures corresponding to a new text document;
- means for developing a report by referencing the software elements created from the one or more text documents to form a structure of the report and retrieving data from one or more sources to represent one or more values within the report; and
- means for generating a relationship between the data from the one or more sources and the one or more values to be placed within the report.

30. A computer-readable medium including instructions for controlling a processor to perform a method for developing a report, the method comprising the steps of:

- receiving one or more text documents;
- creating software elements having a format with a hierarchal relationship between the software elements based on the one or more text documents;
- creating a second hierarchical relationship between the software elements;

restructuring the first hierarchical relationship and the second hierarchical relationship into software structures corresponding to a new text document;

developing a report by referencing the software elements created from the one or more text documents to form a structure of the report and retrieving data from one or more sources to represent one or more values within the report; and

generating a relationship between the data from the one or more sources and the one or more values to be placed within the report.

31. The computer-readable medium of claim 30, wherein creating the software elements from the one or more text documents includes representing the software elements in a Numerator Document Object Model (NDOM).

32. The computer-readable medium of claim 30, wherein creating the software elements includes creating software elements from one or more XBRL documents.

33. The computer-readable medium of claim 30, wherein creating the software elements having the format with the hierarchal relationship includes interpreting tags included in the one or more text documents.

34. The computer-readable medium of claim 30, further comprising manipulating the software elements by browsing, editing, loading, and storing the software elements.

36. The computer-readable medium of claim 30, further comprising developing the report from one or more templates, which contain data that is directly inserted into the report and instructions enabling data from the one or more sources to be inserted into the report.

37. The computer-readable medium of claim 35, wherein the relationship is generated through a “drag and drop” process.

38. The computer-readable medium of claim 30, further comprising modifying the software elements to create a new combination of software elements representative of a new text document.

39. The computer-readable medium of claim 30, further comprising modifying the software elements by editing one or more parameters associated with the software elements.

40. The computer-readable medium of claim 30, further comprising transforming the software elements to new software elements for importing into an RDL system.

41. The computer-readable medium of claim 40, wherein transforming the new software elements includes retrieving a tag associated with each of the software elements in a dictionary and invoking a translation routine associated with the tag.

42. A data processing system for developing a report, comprising:

a parser that receives one or more text documents and creates software elements having a format with a hierarchal relationship between the software elements based on the one or more text documents;

a manager that manipulates the software elements, provides for the creation of a second hierarchical relationship between the software elements, and provides for the restructuring of the first hierarchical relationship and the second hierarchical relationship into software structures corresponding to a new text document;

an editor that develops a report by referencing the software elements created from the one or more text documents to form a structure of the report; and

a mapper that retrieves data from one or more sources to represent one or more values within the report and generates a relationship between the retrieved data and the one or more values within the report.

43. The data processing system of claim 42, wherein the format with the hierarchal relationship between the software elements is a Numerator Document Object Model (NDOM).

44. The data processing system of claim 42, wherein one or more text documents are XBRL documents.

45. The data processing system of claim 42, wherein the parser creates the software elements with the format with the hierarchal relationship by interpreting tags included in the one or more text documents.

46. The data processing system of claim 42, wherein the manager manipulates the software elements by browsing, editing, loading, and storing the software elements.

47. The data processing system of claim 42, wherein the report is developed from one or more templates, which contain data that is directly inserted into the report and instructions enabling data from the one or more sources to be inserted into the report.

48. The data processing system of claim 47, wherein the one or more templates provides instructions to the mapper to retrieve the data that is directly inserted into the report and data from local or remote sources.

49. The data processing system of claim 42, wherein the report and the one or more sources are linked through a "drag and drop" process.

50. The data processing system of claim 42, wherein the editor provides for the software elements to be modified to create a new combination of software elements representative of a new text document.

51. The data processing system of claim 42, wherein the editor provides for modification of one or more parameters associated with the software elements.

52. The data processing system of claim 42, wherein the software elements are transformed to new software elements and are imported into an RDL system.

53. The data processing system of claim 52, wherein the software elements are transformed to the new software elements by retrieving a tag associated with each of the software elements in a dictionary and invoking a translation routine associated with the tag.

54. A method for data processing, comprising:

- receiving one or more text documents;
- creating software elements having a format with a hierarchal relationship between the software elements based on the one or more text documents;
- manipulating the software elements;
- creating a second hierarchical relationship between the software elements;
- restructuring the first hierarchical relationship and the second hierarchical relationship into software structures corresponding to a new text document;
- developing a report by referencing the software elements created from the one or more text documents to form a structure of the report;
- generating a relationship between data from one or more sources and one or more values to be placed within the report; and

retrieving data from the one or more sources to represent the one or more values within the report.

55. The method of claim 54, wherein creating the software elements from the one or more text documents includes representing the software elements in a Numerator Document Object Model (NDOM).

56. The method of claim 54, wherein creating the software elements from the one or more text documents includes creating software elements from one or more XBRL documents.

57. The method of claim 54, wherein creating the software elements having the format with the hierarchal relationship includes interpreting tags included in the text documents.

58. The method of claim 54, further comprising developing the report from one or more templates, which contain data that is directly inserted into the report and instructions enabling data from the one or more sources to be inserted into the report.

59. The method of claim 54, further comprising modifying the software elements to create a new combination of software elements representative of a new text document.

60. The method of claim 54, further comprising modifying the software elements by editing one or more parameters associated with the software elements.

61. The method of claim 54, further comprising transforming the software elements to new software elements for importing into an RDL system.

62. A data processing system, comprising:
a parser that:
receives one or more text documents,
interprets tags included in the one or more text documents to create software elements, and
determines the hierarchy of the software elements within a structure representative of the one or more text documents; and
a manager that:
provides for the creation of a second hierarchy of the software elements, and
provides for the restructuring of the first hierarchy and the second hierarchy into software structures corresponding to a new text document.

63. The data processing system of claim 62, wherein the structure is a Numerator Document Object Model (NDOM).

64. The data processing system of claim 62, wherein the one or more text documents are XBRL documents.

IX. Evidence Appendix to Appeal Brief Under Rule 41.37(c)(1)(ix)

None.

X. Related Proceedings Appendix to Appeal Brief Under Rule 41.37(c)(1)(x)

None.